(19) Deutsches Patent- und Markenamt

(10) **DE 20 2013 012 491 U1** 2017.03.02

(12) # Gebrauchsmusterschrift

(21) Aktenzeichen: **20 2013 012 491.1**
(22) Anmeldetag: **28.02.2013**
(67) aus Patentanmeldung: **EP 13 71 0210.9**
(47) Eintragungstag: **24.01.2017**
(45) Bekanntmachungstag im Patentblatt: **02.03.2017**

(51) Int Cl.: ***G06F 9/54*** (2006.01)

(30) Unionspriorität:
**13/409,651** **01.03.2012** **US**

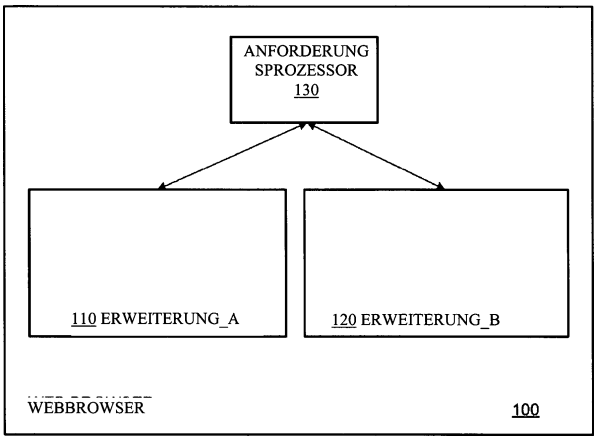
(73) Name und Wohnsitz des Inhabers:
**GOOGLE INC., Mountain View, Calif., US**

(74) Name und Wohnsitz des Vertreters:
**Betten & Resch Patent- und Rechtsanwälte PartGmbB, 80333 München, DE**

**Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen**

(54) Bezeichnung: **Quer-Erweiterungsnachrichtenübermittlung unter Verwendung eines Browsers als Intermediär**

(57) Hauptanspruch: Nicht flüchtiges computerlesbares Speichermedium, das Befehle aufgezeichnet und gespeichert hat, die, wenn sie von einem Computergerät ausgeführt werden, das Computergerät zu Folgendem veranlassen:
dem Empfangen einer Anfrage einer ersten Browser-Erweiterung über eine Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen durch den Webbrowser, worin ein Empfänger der Anfrage eine zweite Browser-Erweiterung ist, ermittelt durch die Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen basierend auf einer angefragten Funktionalität, die in der Anfrage der ersten Browser-Erweiterung beinhaltet ist;
dem Senden, unter Verwendung der Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen, der Anfrage an die zweite Browser-Erweiterung, wenn die angefragte Funktionalität in der zweiten Browser-Erweiterung verfügbar ist; und
dem Senden einer Fehlermeldung an die erste Browser-Erweiterung mithilfe der Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen, wenn die angefragte Funktionalität nicht verfügbar ist.



## Beschreibung

### VERWEISE

**[0001]** Unter Schutz gestellt werden und Gegenstand des Gebrauchsmusters sind dabei, entsprechend den Vorschriften des Gebrauchsmustergesetzes, lediglich Vorrichtungen wie in den beigefügten Schutzansprüchen definiert, jedoch keine Verfahren. Soweit nachfolgend in der Beschreibung gegebenenfalls auf Verfahren Bezug genommen wird, dienen diese Bezugnahmen lediglich der beispielhaften Erläuterung der in den beigefügten Schutzansprüchen unter Schutz gestellten Vorrichtung oder Vorrichtungen.

### TECHNISCHES GEBIET

**[0002]** Diese Anmeldung bezieht sich im Allgemeinen auf Webbrowser-Technologie.

### HINTERGRUND

**[0003]** Die Nutzung des World Wide Web nimmt weiter zu, wie auch die Menge und Variation des Contents, die den Nutzern zugänglich ist. Nutzer des World Wide Web benutzen für gewöhnlich auf webfähigen Computergeräten implementierte Browser (Webbrowser) für den Zugang zum Content. Solche Geräte beinhalten Personalcomputer, Laptop Computer, Smartphones und Mobiltelefone, neben einer Anzahl anderer möglicher Geräte. Solche Webbrowser sind konfiguriert um programmatischen Code zu lesen und diesen Code als Websites zu rendern, was das Rendern von beidem audio- und visuellem Content enthalten in verschiedenen Mediadateien beinhalten kann (z. B. Image, Video- und Audiodateien), wie auch Durchführen anderer im programmatischen Code definierter Funktionen. Websites werden im Allgemeinen durch die Nutzung von Programmiersprachen, wie HTML5 und JavaScript, neben einer Anzahl anderer verfügbarer Programmiersprachen, implementiert.

**[0004]** Manche Browser erlauben Nutzern Add-Ons (oder Erweiterungen) zum Browser zu installieren, wo diese Erweiterungen dem Browser Funktionalitäten hinzufügen und als integrierter Teil des Browsers fungieren. Eine Erweiterung könnte einem Nutzer beispielsweise Zugang zu ihren zusätzlichen Funktionalitäten durch Modifizieren einer Benutzeroberfläche (UI) des Browsers verschaffen. So kann beispielsweise einem Browser eine Erweiterung zur Wetterprognose, wobei die Erweiterung einfachen Zugang zu Wetterinformationen bietet, durch Hinzufügen eines Icons oder eines Buttons zur Benutzeroberfläche des Browsers hinzugefügt werden. Ein Nutzer kann dann mit dem Button oder Icon der Erweiterung interagieren (z. B. durch Anklicken oder Darüberfahren mit einem Zeigegerät), um Wetterinformationen zu erhalten, statt zu einer wetterbezogenen Website zu navigieren, um Wetterinformationen zu erhalten.

**[0005]** Browser-Erweiterungen werden im Allgemeinen durch die Nutzung eines programmatischen Codes implementiert, der durch die gleichen Programmiersprachen, die für die Implementierung von Websites genutzt werden, geschrieben wird, wie etwa JavaScript. Aus der Perspektive eines Browsers funktionieren Erweiterungen effektiv wie Websites, die nach dem Installieren ein integrierter Teil des Browsers sind. Durch Installieren von Erweiterungen, die für den Nutzer von Interesse sind, kann er bzw. sie effektiv einen benutzerdefinierten Browser, der die Funktionalitäten der von ihm für die Installation gewählten Erweiterungen beinhaltet, generieren. So kann beispielsweise ein Nutzer Erweiterungen benutzen, um synthetisierten Text-to-Speech wiederzugeben, zum Beispiel beim Lesen eines Buches über einen Webbrowser. Es kann für Erweiterungen nützlich sein, Nachrichten auszutauschen, während die Privatsphäre eines Nutzers gewahrt bleibt, und während es einem Webbrowser erlaubt wird, Fehler ordnungsgemäß zu handeln.

### ZUSAMMENFASSUNG

**[0006]** In einem allgemeinen Aspekt beinhaltet ein Verfahren, um Nachrichten mithilfe eines Webbrowsers eines Computergerätes zwischen mindestens zwei Browser-Erweiterungen zu übertragen, das Empfangen einer Anfrage einer ersten Browser-Erweiterung über eine Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen durch den Webbrowser, worin ein Empfänger der Anfrage eine zweite Browser-Erweiterung ist, ermittelt durch die Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen basierend auf einer angefragten Funktionalität, die in der Anfrage der ersten Browser-Erweiterung beinhaltet ist. Das Verfahren beinhaltet das Senden, unter Verwendung der Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen, die Anfrage an die zweite Browser-Erweiterung, wenn die angefragte Funktionalität in der zweiten Browser-Erweiterung verfügbar ist, und das Senden einer Fehlermeldung zur ersten Browser-Erweiterung mithilfe der Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen, wenn die angefragte Funktionalität nicht verfügbar ist.

**[0007]** In einem weiteren allgemeinen Aspekt hat ein nicht flüchtiges computerlesbares Datenspeichermedium darauf Instruktionen gespeichert und aufgenommen, die, wenn sie durch ein Computergerät ausgeführt werden, das Computergerät dazu veranlassen, über eine Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen eine Anfrage von einer ersten Browser-Erweiterung zu empfangen, bereitgestellt durch einen Webbrow-

ser des Computergerätes, worin ein Empfänger der Anfrage eine zweite Browser-Erweiterung ist, ermittelt von der Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen, basierend auf einer angefragten Funktionalität, enthalten in der Anfrage der ersten Browser-Erweiterung; das Senden – über die Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen – der Anfrage an die zweite Browser-Erweiterung, wenn die angefragte Funktionalität in der zweiten Browser-Erweiterung verfügbar ist; und das Senden einer Fehlermeldung über die Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen an die erste Browser-Erweiterung, wenn die angefragte Funktionalität nicht verfügbar ist.

**[0008]** Gemäß einem weiteren allgemeinen Aspekt, beinhaltet eine Apparatur einen Speicher und einen Prozessor operierend gekoppelt an den Speicher. Der Prozessor kann konfiguriert werden um Code auszuführen um eine Anfrage von einer ersten Browser-Erweiterung zu empfangen von einer Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen bereitgestellt durch einen Webbrowser des Computergerätes, woraufhin ein Empfänger der Anfrage eine zweite Browser-Erweiterung ist determiniert durch die Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen basierend auf einer angefragten Funktionalität enthalten in der Anfrage der ersten Browser-Erweiterung; Senden, die Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen nutzend, die Anfrage an die zweite Browser-Erweiterung wenn die angefragte Funktionalität verfügbar ist in der zweiten Browser-Erweiterung; und eine Fehlermeldung senden an die erste Browser-Erweiterung die Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen nutzend wenn die angefragte Funktionalität nicht verfügbar ist.

**[0009]** Die Einzelheiten einer oder mehrerer Implementierungen sind in den begleitenden Zeichnungen und der Beschreibung unten dargelegt. Andere Features anhand der Beschreibung und Zeichnungen sowie anhand der Ansprüche ersichtlich werden.

**[0010]** Ein System und/oder Verfahren für die Nachrichtenübermittlung zwischen Browser-Erweiterungen unter Verwendung des Browsers als Intermediär, im Wesentlichen, wie gezeigt in und/oder beschrieben in Verbindung mit Figuren, wie in den Ansprüchen vollständiger dargelegt.

## KURZBESCHREIBUNG DER ZEICHNUNGEN

**[0011] Fig.** 1 ist ein Blockdiagramm, das einen exemplarischen Webbrowser veranschaulicht, welcher die Erweiterungs-Nachrichtenübermittlung mithilfe des Browsers als Intermediär implementiert.

**[0012] Fig.** 2 ist ein Blockdiagramm, welches eine Inter-Erweiterungsnachrichtenübermittlungsarchitektur gemäß einer exemplarischen Implementierung veranschaulicht.

**[0013] Fig.** 3 ist eine exemplarische Benutzeroberfläche eines Webbrowsers in Übereinstimmung mit der exemplarischen Implementierung, die beschrieben ist im Hinblick auf **Fig.** 2.

**[0014] Fig.** 4 ist ein Flussdiagramm, das ein Verfahren zur Erweiterungs-Nachrichtenübermittlung unter Verwendung des Browsers als Intermediär veranschaulicht.

**[0015] Fig.** 5 ist ein Flussdiagramm, welches ein Verfahren für die Erweiterungs-Nachrichtenübermittlung über einen Browser als Intermediär veranschaulicht, gemäß einer exemplarischen Implementierung, wie oben gemäß **Fig.** 2 und **Fig.** 3 erläutert.

**[0016] Fig.** 6 ist ein Diagramm, welches ein Computergerät und ein mobiles Computergerät veranschaulicht, die nach einer exemplarischen Ausführungsform genutzt werden können, um die hierin beschriebenen Techniken zu implementieren.

**[0017]** Ähnliche Referenzsymbole in den verschiedenen Zeichnungen verweisen auf ähnliche Elemente.

## AUSFÜHRLICHE BESCHREIBUNG

**[0018]** Ein Webbrowser, ausgeführt durch ein Client-Gerät, kann Code (z. B. HTML Code) von einem Remote-Server (z. B. einem Remote-Server, der eine Website hostet) empfangen und den empfangenen Code zum Nutzen eines Nutzers des Client-Gerätes auf dem Client-Gerät ausführen.

**[0019]** In verschiedenen Implementierungen, kann der Webbrowser eine oder mehrere Webanwendungen beinhalten oder damit verbunden sein. In diesem Zusammenhang, kann eine „Webanwendung" konfiguriert sein, um eine einzelne Aufgabe oder mehrere Aufgaben für den Nutzer durchzuführen. In einer solchen Implementierung, kann die Webanwendung konfiguriert sein, um durch den Webbrowser ausgeführt oder interpretiert zu werden. Das wird verglichen mit den nativen Anwendungen, die maschinenausführbaren Code beinhalten und konfiguriert sind, um direkt von einem Prozessor oder durch das Betriebssystem des Client-Gerätes ausgeführt zu werden, wohingegen eine Webanwendung ohne die Hilfe des Webbrowsers zur Ausführung oder Anzeige unfähig sein kann. Somit können Webanwendungen innerhalb eines Browsers mit einer dedizierten Benutzeroberfläche ausgeführt werden, und können so Funktionalität und eine Erfahrung bieten die reichhaltiger ist und interaktiver als eine eigenständige Web-

site aber weniger umständlich und monolithisch als eine Desktopanwendung. Beispiele für Webanwendungen beinhalten Spiele, Foto-Editoren und Videoplayer, die innerhalb des Browsers ausgeführt werden.

**[0020]** Webanwendungen können „gehostete Webanwendungen", „installierbare Webanwendungen" oder „gepackte Webanwendungen" sein. Gehostete Webanwendungen können wenigstens einen Teil einer Website beinhalten, welche selbst Websites beinhaltet, sowie einige Metadaten, die besonders relevant für die Webanwendung oder für den Nutzer der Webanwendung sein können, um der Webanwendung zu erlauben, einige spezielle Funktionalitäten auszuführen.

**[0021]** Installierbare Webanwendungen sind eine Möglichkeit, um einen Browser bestimmte Websites wie Anwendungen behandeln zu lassen. Viele installierbare Webanwendungen sind gehostete Webanwendungen mit zusätzlichen Metadaten (wie etwa eine kleine Manifestdatei welche die Anwendung beschreibt). Gepackte Webanwendungen sind eine Art von installierbaren Webanwendungen. Gepackte Webanwendungen können als Webanwendungen angesehen werden, deren Code gebündelt ist, sodass der Nutzer den gesamten Content der Webanwendung zur Ausführung durch den Browser herunterladen kann. Eine gepackte Webanwendung benötigt u. U. keinen Netzwerkzugang, um ihre Funktionalität für den Nutzer durchzuführen, und kann stattdessen durch den Browser erfolgreich lokal auf dem Computergerät ohne Zugang zu einem Netzwerk ausgeführt werden. Bei gepackten Webanwendungen besteht die Möglichkeit, Erweiterungs-Programmierschnittstellen (APIs) zu nutzen, wodurch gepackte Apps die Art ändern können, wie sich der Browser verhält oder aussieht.

**[0022]** In verschiedenen Beispielen kann der Webbrowser eine oder mehrere Browser-Erweiterungen beinhalten oder für eine Interaktion damit konfiguriert sein. In diesem Zusammenhang, kann eine „Browser-Erweiterung" eine oder mehrere gepackte oder als ein definierbares Ganzes zusammengefasste Websites beinhalten, und so konfiguriert sein, dass sie die Funktionalität auf den Webbrowser ausdehnt. Wie hierin beschrieben, sind „Browser-Erweiterungen" kleine Softwareprogramme, welche die Funktionalität eines Webbrowsers modifizieren und erweitern können. Sie können unter Verwendung von Webtechnologien, wie etwa HTML, JavaScript und CSS, geschrieben sein. Erweiterungen können wenige oder keine Benutzeroberflächen haben. Erweiterungen können ein Benutzeroberflächensymbol bereitstellen oder andere Benutzeroberflächen, die den „Chrome" eines Browsers modifizieren können, wie hierin definiert, sodass sie einen Bereich beinhaltet, der sich außerhalb einer Website befindet und im Browser angezeigt wird (wie etwa die Grenzen eines Webbrowser-Fensters, welche das Browserfenster, Menüs, Symbolleisten und Bildlaufleisten beinhalten).

**[0023]** Somit stellen Browser-Erweiterungen einem Browser zusätzliche Funktionalitäten bereit, im Gegensatz zu Webanwendungen gibt es jedoch im Allgemeinen nur wenige oder keine Benutzerschnittstellenkomponenten für die durch die Erweiterung bereitgestellten Funktionalitäten. Stattdessen erweitern Browser-Erweiterungen die Funktionalität des Browsers und der darin betrachteten Websites. Browser-Erweiterungen können beispielsweise die Funktionalität des Browsers durch Hinzufügen eines neuen Buttons zur Adressleiste erweitern, wie etwa ein permanent präsenter Wechselkursrechner. Solche Buttons können auch auf die aktuell betrachtete Website angewendet werden – so kann beispielsweise das Klicken auf den Wechselkursrechner-Button alle Preise auf der Website, die einem Benutzer angezeigt werden, in eine vom Nutzer ausgewählte Währung umrechnen. In einem anderen Beispiel kann eine Erweiterung installiert sein, sodass, wenn ein Nutzer mit einem Cursor über ein Vorschaubild fährt, so lange auf einer vom Browser ausgeführten Website eine größere Version des Bildes angezeigt wird, bis der Nutzer den Cursor vom Bild wegbewegt. In einem anderen Beispiel kann eine Erweiterung installiert sein, um einen Button „Dies versenden" neben jedem Link auf jeder Seite einzubetten. Im Vergleich zu Webanwendungen sind Erweiterungen Website- und Webanwendungsübergreifend. Erweiterungen sind gewöhnlich über alle Websites hinweg wirksam (obwohl manche Site-spezifisch sind). Webanwendungen vereinen sich nicht in dieser Weise mit anderen Anwendungen. Webanwendungen werden eher als Einzelanwendungen ausgeführt, wie jede reguläre Website. Gepackte Webanwendungen und installierbare Webanwendungen unterscheiden sich ebenfalls von Browser-Erweiterungen, weil sie sehr unterschiedliche Benutzeroberflächen präsentieren.

**[0024]** „Gepackte Webanwendungen" können im Gegensatz zu Browser-Erweiterungen können aussehen und sich anfühlen wie reguläre Webanwendungen, mit einem Großbild-Design und funktionsreicher Benutzeroberfläche. Sowohl Erweiterungen als auch gepackte Webanwendungen können all ihre Dateien in eine einzelne Datei bündeln, die ein Nutzer herunterlädt und installiert. Dieses Bündeln bedeutet, dass, im Unterschied zu gewöhnlichen Webanwendungen, Erweiterungen und gepackte Webanwendungen nicht von Content im Web abhängig sein müssen. Browser-Erweiterungen, gepackte Webanwendungen und installierbare Webanwendungen können durch einen vertrauenswürdigen Online-Shop oder über einzelne Websites vertrieben werden.

**[0025]** **Fig.** 1 ist ein Blockdiagramm, das einen Webbrowser **100** veranschaulicht, welcher eine Inter-Erweiterungs-Nachrichtenübermittlung unter Verwendung des Browsers als Intermediär implementiert, gemäß einer exemplarischen Implementierung. Der Browser **100** kann in ein Computergerät implementiert sein, wie etwa die beschriebenen und dargestellten Computergeräte in Zusammenhang mit **Fig.** 6 unten. Wie **Fig.** 1, der Browser **100** kann genutzt werden, um einen ersten Erweiterungsprozess, Erweiterung_A **110**, zu implementieren, sowie einen zweiten Erweiterungsprozess, Erweiterung B **120**. Im Browser **100** sind die Erweiterung A **110** und die Erweiterung B **120** so implementiert dass sie von den Funktionalitäten her voneinander isoliert sind, um so die Risiken zu reduzieren, die mit der Ausführung von Erweiterungen in einer offenen Ausführungsumgebung verbunden sind. In diesem Beispiel kann jeder der Erweiterungsprozesse **110** und **120** über einen Anforderungsprozessor **130** des Browsers **100** miteinander kommunizieren. Erweiterungsprozesse **110** und **120** können miteinander kommunizieren ohne voneinander zu wissen, zum Beispiel ohne irgendeine zugeordnete eindeutige Kennung der Erweiterung zu kennen.

**[0026]** Den Anforderungsprozessor **130** nutzend, können Erweiterung_A **110** und Erweiterung_B **120** ein Nachrichtenprotokoll implementieren das ihnen eine synchronisierte Operation erlaubt, während sie ihre funktionale Isolation beibehalten und ohne die Identifikation oder das Wissen voneinander zu benötigen. Anforderungsprozessor **130** kann als eine oder mehrere öffentliche Schnittstelle(n) für die Programmierung von nachrichtenverarbeitenden Programmen (API) operieren, die konfiguriert ist, um Nachrichten zwischen Erweiterungen zu vermitteln. Die APIs können synchronisiert sein oder sie können asynchron sein, sodass sie unmittelbar und ohne darauf zu warten, dass die Operation endet, zurückkehren können. Zwar sind nur zwei Erweiterungsprozesse **110** und **120** und ein Anforderungsprozessor **130** in **Fig.** 1 dargestellt und beschrieben, es wird jedoch anerkannt, dass weitere Erweiterungen oder Anforderungsprozessoren im Browser **100** implementiert sein können, und dass die hierin beschriebenen Techniken gleichermaßen für jene weiteren Erweiterungen und Anforderungsprozessoren gelten würden.

**[0027]** **Fig.** 2 ist ein Blockdiagramm, welches eine Inter-Erweiterungsnachrichtenübermittlungsarchitektur gemäß einer exemplarischen Implementierung veranschaulicht. Die exemplarische Implementierung, die in Bezug auf **Fig.** 2 dargestellt und erläutert ist, dient nur zu illustrativen Zwecken, und ist nicht dazu gedacht, die beschriebenen Verfahren und Systeme einzuschränken. Obwohl die exemplarische Implementierung eine Webanwendung nutzt um ein Buch laut vorzulesen, wird ein Fachmann verstehen dass das beschriebene System und Methoden auf jede Art von Webanwendungen oder Browser-Erweiterungen Anwendungen finden können, zum Beispiel wie oben in Zusammenhang mit **Fig.** 1 beschrieben. Als solches, sind die offengelegten Systeme und Methoden nicht einfach auf Buchvorleseanwendungen oder Sprachbearbeitungsanwendungen und Browser-Erweiterungen limitiert wie beschrieben in Zusammenhang mit **Fig.** 2. Als andere Beispiele, die beschriebenen Systeme und Methoden können implementiert werden Webanwendungen und Browser-Erweiterungen nutzend bezogen auf Spiele, Videos, Wetter, Finanzen, Produktivität, Soziales, Business, Unterhaltung, News, Computerprogrammierung, oder jede andere Webanwendung, Browser-Erweiterung, oder Kombination davon. Jede Art und Anzahl von Webanwendung(en) und/oder Browser-Erweiterung(en) kann genutzt werden um die hierin beschriebenen Systeme und Methoden zu implementieren.

**[0028]** Wie in **Fig.** 2 ersichtlich, beinhaltet Browser **100** eine „Book Reader”-Webanwendung **220**, welche eine Webanwendung repräsentieren kann, die von einem Nutzer eines Computergerätes, welcher den Browser **100** ausführt, installiert wurde. Book Reader-Webanwendung **220** kann eine Fähigkeit beinhalten, ein Buch laut und mithilfe einer Text-to-Speech (TTS) Api **204** vorzulesen. In bestimmten Implementierungen, können Browser native Unterstützung leisten für Sprachen, welche Sprachsynthese-Fähigkeiten nutzen, die durch das operative System bereitgestellt sind. Alternativ oder zusätzlich kann ein Nutzer Browser-Erweiterungen installieren, die sich selbst als alternative Sprach-Engines, wie unten ausführlicher beschrieben, registrieren.

**[0029]** Im Beispiel, dargestellt in **Fig.** 2, kann Browser **100** auch Erweiterungen beinhalten, die Text-to-Speech Engines implementieren. Die TTS ENGINE API **202** erlaubt einer Browser-Erweiterung, Text-to-Speech Engines zu implementieren. Zum Beispiel kann in manchen Implementierungen, wenn sich eine Browser-Erweiterung unter Verwendung der TTS ENGINE API **202** registriert, die Erweiterung Ereignisse empfangen, die eine Äußerung beinhalten, welche gesprochen werden soll, sowie andere Parameter, wenn irgendeine Browser-Erweiterung oder gepackte Webanwendung die TTS API **204** zur Sprachgenerierung nutzt. Die Browser-Erweiterung kann dann eine verfügbare Webtechnologie nutzen, um die Sprache zu synthetisieren und auszugeben, und kann Ereignisse zur Ausrufungsfunktion zurücksenden, um den Status zu melden. „Ereignisse” können als Objekte visualisiert werden, welche die Benachrichtigung einer Webbrowser-Erweiterung (oder Webanwendung) gestatten, wenn etwas passiert.

**[0030]** In manchen exemplarischen Implementierungen, kann jedes Ereignis eine Ereignisart, einen Zei-

chenindex einer aktuellen Sprache, bezogen auf die Äußerung, und für Fehler-Ereignisse eine optionale Fehlermeldung beinhalten. Die Ereignisarten können Folgendes beinhalten: 'Start': Die Engine hat das Aussprechen der Äußerung gestartet; 'Wort': eine Wortgrenze wurde erreicht; 'Satz': eine Satzgrenze wurde erreicht; 'Marker': ein Marker wurde erreicht; 'Ende': Die Engine ist mit der Aussprache der Äußerung fertig; 'Unterbrochen': diese Äußerung wurde durch eine andere Ausspracheanforderung unterbrochen() oder angehalten() und nicht beendet; 'Aufgehoben': Diese Äußerung wurde in die Warteschlange gestellt, dann jedoch von einem anderen Aufruf zum Sprechen() oder Anhalten() storniert und daher wurde nie mit dem Aussprechen begonnen; und 'Fehler': Ein enginespezifischer Fehler ist eingetreten und die Äußerung kann nicht gesprochen werden.

**[0031]** Im Beispiel, dargestellt in **Fig.** 2, beinhalten die Erweiterungen Alicia **206**, Barry **208** und Charlie **210**, welche mit dem Browser **100** registriert sind (z. B. von einem Nutzer, der die Erweiterungen installiert hat, zum Beispiel in einem Online-Shop). Im Beispiel, dargestellt in **Fig.** 2, kann jede Erweiterung **206**, **208**, **210** Klänge produzieren, wie die Stimme unterschiedlicher Personen, und jede einzelne kann unterschiedliche Audioqualitäten haben. Wie oben dargelegt, kann sich eine Erweiterung, wie die Erweiterungen **206**, **208** und **210**, sich selbst als Sprachengine registrieren. Dadurch kann die Erweiterung manche oder alle Funktionsaufforderungen abfangen, wie etwa Sprache() und Stopp(), und eine alternative Implementierung bereitstellen. Erweiterungen können frei sein, jede verfügbare Webtechnologie zu nutzen, um Sprache in verschiedenen Audioformaten bereitzustellen, einschließlich Audioübertragungen von einem Server, HTML5 Audio, Native Client oder Flash. Eine Erweiterung kann sogar unterschiedliche Handlungen mit den Äußerungen durchführen, z. B., Untertitel in einem Pop-Up-Fenster anzeigen, oder diese als Protokolleinträge an einen Remote-Server senden. Des Weiteren kann eine Erweiterung jede Anzahl an Stimmen spezifizieren.

**[0032]** In einer Implementierung kann die TTS ENGINE API **202** einen Server anrufen, der Sprache rendert und Audio über ein Netzwerk wiedergibt. Eine andere Implementierung generiert Sprache lokal auf dem Computer ohne den Server überhaupt zu involvieren. Auch wenn die TTS ENGINE API **202** einen Server nutzt – die TTS ENGINE API **202** operiert nicht auf einem Server – die TTS ENGINE API **202** operiert dennoch lokal, und doch kann TTS ENGINE API **202** mit einem Remote-Server kommunizieren, um mit der Implementierung von Sprachverarbeitung zu helfen.

**[0033]** In einer exemplarischen Implementierung hat ein Nutzer anfänglich eine Book Reader-Webanwendung **220**, Alicia **206**, und Barry **208** im Browser **100** des Nutzers installiert. Obwohl Book Reader-Webanwendung **220** eine Webanwendung ist und Alicia **206** und Barry **208** Erweiterungen sind, sind die Unterschiede geringfügig und werden nur zu Veranschaulichungszwecken genutzt. Die hierin beschriebenen Beispiele können mit jeder Vielfalt von Kombinationen von Webanwendungen und/oder Erweiterungen implementiert sein. Im Beispiel, dargestellt in **Fig.** 2, der Nutzer kann alle drei Formen von einem vertrauenswürdigen Online-Shop oder einem digitalen Marktplatz installiert haben. Wenn Alicia **206** und Barry **208** im Browser **100** geladen werden, können sich Alicia **206** und Barry **208** selbst mit dem Browser **100** unter Verwendung der TTS ENGINE API **202** registrieren, sodass Browser **100** sie als Sprachengines kennt.

**[0034]** **Fig.** 3 veranschaulicht eine exemplarische Benutzeroberfläche eines Webbrowsers gemäß der exemplarischen Implementierung, beschrieben in Bezug auf **Fig.** 2. Wie **Fig.** 3, ein Nutzer kann eine Browser-Benutzeroberfläche **310** einschließlich der Tabs **312** und **314** sehen. Jeder Tab in der Browser-Anwendung kann einen unterschiedlichen Prozess nutzen, um die gesamte Browser-Anwendung vor Fehlern (Bugs) und Glitches in einer Rendering-Engine zu schützen, die mit der Browser-Anwendung assoziiert ist. Jeder Tab kann verschiedene Prozesse ausführen, von denen jeder einen Lesen-/Schreibzugang zu wenigstens einem Teil der Speicher des Computergerätes hat, auf dem die Browser-Anwendung ausgeführt wird. Der Zugang kann von einem Rendering-Engine-Prozess zu einem anderen Rendering-Engine-Prozess sowie in Bezug auf den Rest des Systems eingeschränkt sein. Auf diese Art kann ein Prozess, der für eine Webanwendung oder für eine Browser-Erweiterung ausgeführt wird, von anderen Prozessen, die für andere Webanwendungen oder für andere Browser-Erweiterungen ausgeführt werden, isoliert und separiert sein.

**[0035]** In manchen Implementierungen können Tabs, wie Tabs **312** und **314**, getrennte Prozesse ausführen, wobei jeder seinen eigenen Speicher und seine eigene Kopie von globalen Datenstrukturen hat. Getrennte Prozesse rendern getrennte Tabs, welche ebenfalls getrennte JavaScript Threads generieren können. Ein Tab (z. B., **312**) kann beschäftigt sein, während alle anderen Tabs in Benutzung sind. Wenn ein Fehler beim Rendern eines Tabs auftritt (z. B. Tab **312**), kann es sein, dass die anderen Tabs nicht betroffen sind, oder aber die gesamte Browser-Anwendung kann abstürzen.

**[0036]** Book Reader-Webanwendung **220** kann nach dem Starten eine grafische Benutzeroberfläche **320** darstellen, die im Tab **312** der Browser-Benutzeroberfläche **310** angezeigt wird. In einer Implementierung kann es sein, dass ein Nutzer zu Anfang ein Buch mit der Book Reader-Webanwendung **320** zu lesen beginnt, sich dann jedoch das Buch lieber an-

hören möchte, anstatt e zu lesen. Der Nutzer kann durch Auswählen eines grafischen Benutzeroberflächenelements der Book Reader-Webanwendung **320** (z. B. durch Drücken von ”Sprechen” **322**) experimentieren. Die Book Reader-Webanwendung **220** kann eine Funktion aufrufen, wie etwa chrome.tts.speak („Es war eine dunkle und stürmische Nacht”) und es wird damit begonnen, das Buch laut vorzulesen. Im Hintergrund wurde die Anweisung zum Vorlesen an den Browser **100** gesendet. Browser **100** weiß, dass zwei TTS-Erweiterungen registriert sind (d. h. Alicia **206** und Barry **208**, dargestellt in **Fig.** 2), und da in diesem Beispiel keine anderen Voraussetzungen vorgegeben wurden, kann Browser **100** eine davon auswählen (beliebig oder zum Beispiel, die, die zuerst registriert wurde, in diesem Beispiel Alicia **206**), und die Nachricht (z. B., mit TTS API **204** und TTS ENGINE API **202**) an Alicia **206** weiterleiten. Alicia **206** kann die Sprache generieren und das Audio wiedergeben und dann eine Nachricht zurücksenden (z. B. an TTS ENGINE API **202**), wenn sie fertig ist. Browser **100** (z. B. durch TTS ENGINE API **204**, kommunizierend mit API **202**, wie in **Fig.** 2 dargestellt) kann die „fertige” Nachricht zurück an die Book Reader-Webanwendung **220** leiten, damit diese weiß, dass die Seite umgeblättert und auf der nächsten Seite fortgefahren werden muss.

**[0037]** In manchen Implementierungen kann ein Nutzer andere Optionen aus einer Book Reader-Webanwendung **220** auswählen, zum Beispiel mit anderen Stimmen zu experimentieren. In diesen Implementierungen kann der Nutzer beispielsweise die Benutzeroberfläche **320** der Book Reader-Webanwendung **220** öffnen, um die Einstellungen der Webanwendung zu modifizieren und eine Stimme zu wählen, zum Beispiel ein Menü **324** der grafischen Benutzeroberfläche, wie dargestellt in **Fig.** 3. Book Reader-Webanwendung **220** fragt den Browser **100** ab und ermittelt, dass zwei Stimmen im Browser **100** installiert sind: Alicia **206** und Barry **208**. Der Nutzer wählt dieses Mal Barry aus. Wenn der Nutzer jetzt „Sprechen” **322** drückt, ruft Book Reader-Webanwendung **220** chrome.tts.speak ab ((Die Camper waren noch immer schockiert, den toten Körper zu sehen.” {StimmName: ”Barry”}), und der Browser **100** leitet die Sprache diesmal automatisch an Barry **208** weiter (zum Beispiel mithilfe von TTS API **204** und TTS ENGINE API **202**, wie dargestellt in **Fig.** 2).

**[0038]** In manchen Fällen, kann eine Webbrowser-Erweiterung oder Webanwendung Fehler erhalten. Durch die Nutzung eines Browser als Intermediär kann dieser Fehler einwandfrei gehandelt werden. Barry **208** stürzt beispielsweise ab, wenn ein zu langer Satz vorgegeben wird. Weil der Browser **100** der Intermediär ist, kann der Browser **100** (z. B., mit TTS ENGINE API **202** und TTS API **204**), wenn Barry **208** in der Mitte eines langen Satzes abstürzt, eine Nachricht an den Book Reader **220** senden, die besagt, dass der vorhergehende Satz „unterbrochen” wurde. So weiß die Book Reader-Webanwendung **220**, dass sie es nochmal mit dem gleichen Satz versuchen sollte, statt zum nächsten weiterzugehen. Die Book Reader-Webanwendung **220** kann an diesem Punkt auch feststellen, dass Barry **208** nicht mehr länger verfügbar ist, und die Book Reader-Webanwendung **220** kann entweder auswählen den Nutzer für eine andere Stimme abzufragen, oder sie kann sich entscheiden, keine Stimme auszuwählen und den Browser **100** eine beliebige andere auswählen zu lassen.

**[0039]** Nutzer können jede Anzahl an Erweiterungen installieren. In einem Beispiel kann der Nutzer eine weitere Erweiterung (Caitlin **210**) im Browser **100** installieren. Das nächste Mal, wenn die Book Reader-Webanwendung **220** eine Liste möglicher Stimmen abfragt, sieht sie nun auch Caitlin **210**, und der Nutzer kann nun beginnen, Caitlin **210** zu nutzen. Als sich Caitlin **210** registriert hat, hat die Erweiterung nur „Englisch” als unterstützte Sprache aufgeführt, aber Alicia **206** hat sowohl Englisch als auch Spanisch aufgeführt. Das Buch kann innerhalb eines Buches auf eine Kombination von Sprachen treffen, wie etwa einen einzelnen Satz auf Spanisch. In einer solchen Implementierung weist der Book Reader-Webanwendung **220** das Aussprechen eines spanischen Satzes an, der Browser **100** kann diese Anforderung stattdessen automatisch an Alicia **206** weiterleiten, weil Alicia **206** sowohl Englisch als auch Spanisch als unterstützte Sprachen aufgeführt hat. Der nächste Satz ist wieder auf Englisch, sodass der Satz an die Stimme von Caitlin **210** zurückgehen kann.

**[0040]** Das alles ist möglich, weil Browser **100** als ein Intermediär agiert. Book-Reader **220** ruft chrome.tts.speak ab (und andere chrome.tts Methoden TTS API **204** nutzend), und die Erweiterungen **206**, **208**, **210** rufen alle chrome.ttsEngine Methoden ab (TTS ENGINE API **202** nutzend), und der Browser **100** sendet ihnen Nachrichten, damit alles funktioniert. Der Browser **100** stellt einen Mechanismus bereit, damit eine Erweiterung eine Nachricht an eine andere Erweiterung sendet, durch die Festlegung einer eindeutigen ID der Erweiterung. Als Intermediär agierend, erlaubt der Browser **100** Kommunikationen zwischen Webanwendungen und Erweiterungen, die anonymer und privater sind, da Erweiterungen und Webanwendungen keine unnötigen Details voneinander wissen müssen – sie müssen beispielsweise nicht die eindeutigen IDs anderer Erweiterungen kennen. Des Weiteren gilt, dass der Browser **100**, als Intermediär auftretend, weniger anfällig für Fehler ist, weil der Browser **100** die Kommunikation erleichtern und die unangemessene Nutzung der API unterbinden kann sowie Fehlervorfälle einwandfrei handelt.

**[0041]** Weil der Browser als Intermediär handelt (z. B. unter Verwendung von TTS API **204** und TTS ENGINE API **202**), kann ein Unternehmen eine Weban-

wendungen erstellen (wie etwa Book Reader **220**), ohne mit anderen Unternehmen, welche Erweiterungen erstellen, zusammenkommen zu müssen (wie etwa Stimmerweiterungen **206**, **208**, und **210**). Darüber hinaus muss die Book Reader-Webanwendung **220** nicht die eindeutige ID jeder möglichen Stimme kennen. Indem Browser **100** als ein Intermediär agiert, muss die Book Reader-Webanwendung **220** nicht jeder möglichen Stimme eine Nachricht senden und sehen, welche antworten, und die Book Reader-Webanwendung **220** muss nicht einen Browser nach einer Liste aller von einem Nutzer installierten Erweiterungen fragen. Indem Browser **100** als ein Intermediär zwischen Book Reader-Webanwendung **220** und den Erweiterungen **206**, **208**, und **210** agiert, müssen Nutzer nicht der Book Reader-Webanwendung **220** die Erlaubnis erteilen, auf die Listen mit den installierten Erweiterungen zuzugreifen (und folglich potenziell sensible Informationen gegenüber anderen Anwendungen exponieren). Des Weiteren gilt, dass, da der Browser **100** als Intermediär agiert, die Book Reader-Webanwendung **220** eine neue Stimme automatisch nutzen kann, wenn sie vom Nutzer installiert wird. Im Gegensatz dazu wüsste, wenn der Browser **100** nicht als Intermediär auftreten würde, die Book Reader-Webanwendung **220** nicht, dass die Stimme existieren würde, d. h., eine dritte Partei könnte keine neue Stimme generieren, ohne mit allen Sprachwebanwendungen zusammenzukommen und davon zu überzeugen, die neue Stimme als eine Wahl anzubieten.

**[0042]** Weil die Erweiterungen **206**, **208**, und **210** und die Book Reader-Webanwendung **220** nicht direkt miteinander zu kommunizieren brauchen, wenn Barry **208** abstürzt, weiß Book Reader-Webanwendung **220**, wie sie den Satz mit einer anderen Stimme wiederholen kann. Weil der Browser **100** der Intermediär ist, kann er ein konsistentes Interface der Book Reader-Webanwendung **220** präsentieren, sodass jeder Abruf der tts.speak entweder erfolgreich sein wird oder in einem Fehler resultiert. Im Gegensatz dazu könnte, wenn der Browser nicht als Intermediär auftreten würde, eine der Erweiterungen (z. B. **206**, **208** oder **210**) abstürzen oder sich aufhängen, und die Webanwendung (z. B. die Book Reader-Webanwendung **220**) weiß unter Umständen nicht, was das Problem ist.

**[0043]** Ähnlich gilt, dass, wo der Browser **100** als Intermediär fungiert, die Book Reader-Webanwendung **220** keine Erweiterungsfehler handeln muss, und dass nicht jede Stimmerweiterung Fehler in Sprechanwendungen (wie etwa Book Reader-Webanwendung **220**) handeln muss. In manchen Implementierungen, wenn der Browser **100** als ein Intermediär genutzt wird, kann er eine automatische Fehlerkontrolle auf beiden Seiten durchführen. Der Browser **100** kann beispielsweise die Book Reader-Webanwendung **220** daran hindern, zu viele Daten auf einmal zu senden, und TTS Engine API **202** ist es unter Umständen nicht gestattet, Benachrichtigungen zu Äußerungen zu senden, die nicht mehr aktiv sind. Die Benutzeroberfläche, die in **Fig.** 3 dargestellt ist, dient nur zu illustrativen Zwecken. Jede Art von Benutzeroberfläche(n) kann genutzt werden, ohne dabei vom Umfang der hierin beschriebenen Verfahren und Systeme abzuweichen.

**[0044]** **Fig.** 4 ist ein Flussdiagramm, welches ein Verfahren **400** für die Erweiterungs-Nachrichtenübermittlung veranschaulicht, bei der ein Browser als Intermediär dient, gemäß einer exemplarischen Implementierung. Das Verfahren **400** kann mithilfe des Browsers **100** implementiert werden, wie dargestellt in **Fig.** 1 und **Fig.** 2. Es wird allerdings anerkannt, dass andere Konfigurationen und Techniken genutzt werden können, um das Verfahren **400** zu implementieren.

**[0045]** Das Verfahren **400** beinhaltet, in Block **410**, das Empfangen – über eine API, welche innerhalb einer Webbrowser-Anwendung eines Computergerätes (z. B. Anforderungsprozessor **130** oder TTS API **204** des Webbrowsers **100**, wie in **Fig.** 1 und **Fig.** 2 dargestellt) operiert – eine Anfrage nach Informationen von einer Browser-Erweiterung (z. B. die Erweiterungsprozess Erweiterung_A **110**). Das Verfahren **400** beinhaltet des Weiteren, in Block **420**, das Senden – innerhalb des Webbrowsers **100** – der Anfrage zu einer zweiten Browser-Erweiterung (z. B., Extension_B **120**). Die erste Browser-Erweiterung und die zweite Browser-Erweiterung wissen nichts voneinander. In manchen Implementierungen, sind Erweiterungen separat als ein installierbares Dateibündel gepackt, einschließlich einer Manifestdatei und mindestens einer HTML-Datei.

**[0046]** In Block **430** beinhaltet das Verfahren **400** das Empfangen – über die API, welche innerhalb der Webbrowser-Anwendung operiert – einer Antwort von der zweiten Browser-Erweiterung. In Block **440** beinhaltet das Verfahren **400** das automatische Bereitstellen – über die API, welche innerhalb der Webbrowser-Anwendung operiert – der Antwort auf die erste Browser-Erweiterung (z. B. durch den Anforderungsprozessor **130**).

**[0047]** Derart kann eine Webanwendung über eine API eine Anforderung an eine Browser-Erweiterung senden. Wenn die Erweiterung über API auf die Anfrage reagiert, kann die Antwort zurück an die Webanwendung gesendet werden. Wenn die Webanwendung allerdings eine deformierte Anforderung versendet, kann der Browser (z. B., die API) den Fehler handeln und eine Fehlerantwort an die Webanwendung senden, sodass die Browser-Erweiterung keine deformierte Anforderung zu sehen braucht. Gleichermaßen gilt, dass wenn die Browser-Erweiterung eine deformierte Antwort sendet, auf eine Anfra-

ge nicht innerhalb der vorbestimmten Zeit antwortet oder abstürzt, kann der Browser eine Fehlerantwort an die Webanwendung senden, sodass die Webanwendung keine deformierte Anforderung zu sehen braucht.

**[0048]** Weil der Browser als Intermediär agiert, müssen Clients der APIs (in diesem Beispiel, die Webanwendungen und die Browser-Erweiterungen) die Kommunikationsdetails (einschließlich gescheiterter Kommunikationen) nicht handeln. Die Webanwendung und die Browser-Erweiterung können nur die API nutzen, und der Browser kann die Kommunikationsdetails handeln, einschließlich des Fehlerhandlings.

**[0049]** **Fig.** 5 ist ein Flussdiagramm, welches ein Verfahren für die Erweiterungs-Nachrichtenübermittlung über einen Browser als Intermediär veranschaulicht, gemäß einer exemplarischen Implementierung, wie oben gemäß **Fig.** 2 und **Fig.** 3 erläutert. Das Verfahren **500** kann unter Verwendung des Browsers **100** implementiert werden, wie dargestellt in **Fig.** 2. Es wird allerdings anerkannt, dass andere Konfigurationen und Techniken zum Implementieren des Verfahrens **500** genutzt werden können, und dass das Verfahren **500** nur exemplarisch ist und die beschriebenen Systeme und Verfahren nicht auf die Anwendung einer bestimmten Art von API, Webanwendung oder Browser-Erweiterung beschränkt ist.

**[0050]** Das Verfahren **500** beinhaltet, in Block **510**, das Empfangen – über eine Text-to-Speech API, welche innerhalb einer Webbrowser-Anwendung eines Computergerätes (z. B. TTS API **204** wie in **Fig.** 2 dargestellt) operiert – einer angeforderten Sprache von einer ersten Webbrowser-Anwendung (z. B. Book Reader-Webanwendung **202**), die innerhalb der Webbrowser-Anwendung ausgeführt wird. Das Verfahren **500** beinhaltet des Weiteren, in Block **520**, das Senden – über die Text-to-Speech API, welche innerhalb der Webbrowser-Anwendung eines Computergerätes operiert – der Anfrage an eine Sprachengine-API, welche innerhalb der Webbrowser-Anwendung eines Computergerätes (z. B. TTS ENGINE API **202**) operiert. Die Methode **500** enthält weiter, beim Block **530**, automatisiertes Weiterleiten der Anfrage an eine Sprachprovider Browser-Erweiterung (z. B., Erweiterung **206**, **208**, **210**) installiert in der Webbrowser-Anwendung eines Computergerätes. Die Sprachprovider Browser-Erweiterung weiß nichts von der ersten Webanwendung (z. B. Erweiterung **206** weiß nicht von Book Reader-Webanwendung **220**).

**[0051]** Beim Block **540**, beinhaltet die Methode **500** automatisiertes Ausgeben, durch die Browseranwendung des Computergerätes, die angeforderte Sprache in einer ersten Stimme bereitgestellt durch eine Sprachprovider Browser-Erweiterung (z. B., in Alicias Stimme von Alicia **206**). Die Stimme kann ausgegeben werden Book Reader-Webanwendung **220** nutzend, zum Beispiel.

**[0052]** **Fig.** 6 ist ein Diagramm, das ein exemplarisches generisches Computergerät **600** und ein generisches mobiles Computergerät **650** zeigt, die mit den hierin beschriebenen Techniken verwendet werden können. Computergerät **600** soll verschiedene Formen von Digitalcomputern darstellen, zum Beispiel Laptops, Desktops, Workstations, Personal Digital Assistants, Server, Blade Server, Mainframes und andere geeignete Computer. Computergerät **650** soll verschiedene Formen mobiler Geräte, wie Personal Digital Assistants, Mobiltelefone, Smartphones und andere ähnliche Computergeräte, darstellen. Die hier dargestellten Komponenten, ihre Verbindungen und Beziehungen, und ihre Funktionen, sollen nur exemplarisch gelten.

**[0053]** Das Computergerät **600** beinhaltet einen Prozessor **602**, einen Speicher **604**, ein Speichergerät **606**, eine High-Speed-Schnittstelle **608**, die mit dem Speicher **604** und -Erweiterungsanschlüssen **610** verbunden wird, und eine Low-Speed-Schnittstelle **612**, die mit einem Low-Speed-Bus **614** und dem Speichergerät **606** verbunden wird. Alle der Komponenten **602**, **604**, **606**, **608**, **610** und **612** sind mithilfe verschiedener Busse miteinander verbunden und können an einer gemeinsamen Hauptplatine oder auf andere Weise, wie geeignet, angebracht sein. Der Prozessor **602** kann Anweisungen zur Ausführung innerhalb des Computergeräts **600** verarbeiten, einschließlich Anweisungen, die im Speicher **604** oder auf dem Speichergerät **606** gespeichert sind, um grafische Informationen fiür eine GUI auf einem externen Eingabe-/Ausgabegerät anzuzeigen, wie Anzeige **616**, die an die High-Speed-Schnittstelle **608** gekoppelt ist. In anderen Implementierungen können mehrere Prozessoren und/oder mehrere Busse verwendet sein, wie angemessen, zusammen mit mehreren Speichern und Speichertypen. Außerdem können mehrere Computergeräte **600** verbunden sein, wobei jedes Gerät Teile der nötigen Operationen bereitstellt (z. B. als Serverbank, eine Gruppe von Blade Servern oder ein Multiprozessor-System).

**[0054]** Der Speicher **604** zeichnet Informationen im Computergerät **600** auf. In einer Implementierung ist der Speicher **604** ein flüchtiges Speichergerät oder flüchtige Speichergeräte. In einer anderen Implementierung, ist der Speicher **604** ein flüchtiges Speichergerät oder flüchtige Speichergeräte. Der Speicher **604** kann auch eine andere Form von computerlesbarem Medium sein, zum Beispiel ein magnetischer oder optischer Datenträger.

**[0055]** Das Speichergerät **606** ist dafür geeignet, Massenspeicherung für das Computergerät **600** bereitzustellen. In einer Implementierung kann das

Speichergerät **606** ein computerlesbares Medium sein oder enthalten, zum Beispiel ein Diskettengerät, ein Festplattengerät, ein optisches Datenträgergerät oder ein Bandgerät, ein Flash-Speicher oder ein anderes ähnliches Solid-State-Speichergerät oder eine Reihe von Geräten, zum Beispiel Geräte in einem Storage Area Network oder anderen Konfigurationen. Ein Computerprogrammprodukt kann konkret in einem Informationsträger ausgeführt sein. Das Computerprogrammprodukt kann auch Anweisungen enthalten, die, wenn sie ausgeführt werden, ein oder mehrere Verfahren ausführen, wie die oben beschriebenen. Der Informationsträger ist ein computer- oder maschinenlesbares Medium, wie der Speicher **604**, das Speichergerät **606** oder der Prozessorspeicher **602**.

**[0056]** Der High-Speed-Controller **608** verwaltet bandbreitenintensive Vorgänge für das Computergerät **600**, während der Low-Speed-Controller **612** Vorgänge mit niedrigerer Bandbreite verwaltet. Eine solche Zuordnung von Funktionen ist nur exemplarisch. In einer Implementierung, ist der High-Speed-Controller **608** an den Speicher **604**, die Anzeige **616** (z. B. durch einen Grafikprozessor oder -beschleuniger) und an die -Erweiterungsanschlüsse **610** gekoppelt, die verschiedene Erweiterungskarten aufnehmen können (nicht gezeigt). In der Implementierung ist der Low-Speed-Controller **612** an das Speichergerät **606** und an den Low-Speed-Erweiterungsanschluss **614** gekoppelt. Der Low-Speed-Erweiterungsanschluss, der verschiedene Kommunikationsanschlüsse (z. B. USB, B, Ethernet, Funkethernet) beinhalten kann, kann an ein oder mehrere Eingabe-/Ausgabe-Geräte, wie eine Tastatur, ein Zeigegerät, einen Scanner oder ein Netzwerkgerät, wie einen Switch oder Router, z. B. durch einen Netzwerkadapter gekoppelt sein.

**[0057]** Das Computergerät **600** kann in einer Reihe verschiedener Formen implementiert werden, wie in der Figur gezeigt. Zum Beispiel kann es als Standardserver **620**, oder mehrmals in einer Gruppe solcher Server implementiert sein. Es kann auch als Teil eines Rackserversystems **624** implementiert sein. Darüber hinaus kann es in einem Personal Computer, wie Laptop-Computer **622**, implementiert sein. Alternativ können Komponenten von Computergerät **600** mit anderen Komponenten in einem mobilen Gerät kombiniert sein (nicht dargestellt), z. B. Gerät **650**. Jedes dieser Geräte kann eines oder mehrere Computergeräte **600**, **650** enthalten, und ein gesamtes System kann aus mehreren Computergeräten **600**, **650** bestehen, die miteinander kommunizieren.

**[0058]** Das Computergerät **650** beinhaltet neben anderen Komponenten einen Prozessor **652**, Speicher **664**, ein Eingabe-/Ausgabegerät, wie eine Anzeige **654**, eine Kommunikationsschnittstelle **666** und einen Sende-Empfänger **668**. Das Gerät **650** kann auch mit einem Speichergerät ausgestattet sein, zum Beispiel einem Microdrive oder einem anderen Gerät, um zusätzlichen Speicher bereitzustellen. Jede der Komponenten **650**, **652**, **664**, **654**, **666** und **668** ist mithilfe verschiedener Busse miteinander verbunden, und kann an einer gemeinsamen Hauptplatine oder auf andere Weise, wie geeignet, angebracht sein.

**[0059]** Der Prozessor **652** kann Anweisungen im Computergerät **650** ausführen, zum Beispiel Anweisungen, die in Speicher **664** gespeichert sind. Der Prozessor kann als ein Chipsatz von Chips implementiert werden, die separate und mehrere analoge und digitale Prozessoren beinhalten. Der Prozessor kann zum Beispiel für die Koordination der anderen Komponenten des Geräts **650** sorgen, zum Beispiel die Kontrolle von Benutzeroberflächen, Anwendungen, die vom Gerät **650** ausgeführt werden, und die drahtlose Kommunikation durch Gerät **650**.

**[0060]** Der Prozessor **652** kann mit einem Benutzer über Steuerschnittstelle **658** und Displayschnittstelle **656** kommunizieren, die mit einem Display **654** gekoppelt ist. Das Display **654** kann zum Beispiel ein TFT LCD(Thin-Film-Transistor Liquid Crystal Display)- oder ein OLED(organisches Leuchtdioden)-Display oder eine andere angemessene Anzeigetechnologie sein. Die Displayschnittstelle **656** kann eine geeignete Schaltung enthalten, die das Display **654** dazu bringt, einem Benutzer grafische und andere Informationen zu präsentieren. Die Steuerschnittstelle **658** kann Befehle von einem Benutzer empfangen und sie für die Sendung an Prozessor **652** umwandeln. Zusätzlich kann eine externe Schnittstelle **662** Kommunikation mit dem Prozessor **652** bereitstellen, zum Beispiel, um Nahbereichskommunikation des Geräts **650** mit anderen Geräten zu ermöglichen. Die externe Schnittstelle **662** kann zum Beispiel in einigen Implementierungen, eine kabelgebundene Kommunikation bereitstellen, oder in anderen Implementierungen eine drahtlose Kommunikation, und es können auch mehrere Schnittstellen verwendet werden.

**[0061]** Der Speicher **664** zeichnet Informationen im Computergerät **650** auf. Der Speicher **664** kann als eines oder mehrere computerlesbare Medien, flüchtige Speichergeräte oder nicht flüchtige Speichergeräte implementiert sein. Erweiterungsspeicher **674** kann ebenfalls bereitgestellt und mit dem Gerät **650** über Erweiterungsschnittstelle **672** verbunden werden, die zum Beispiel eine SIMM(Single In Line Memory Module)-Kartenschnittstelle umfassen kann. Dieser Erweiterungsspeicher **674** kann zusätzlichen Speicherplatz für Gerät **650** bereitstellen oder er kann auch Anwendungen oder andere Informationen für Gerät **650** speichern. Insbesondere kann Erweiterungsspeicher **674** Anweisungen zum Ausführen oder Ergänzen der oben beschriebenen Prozesse enthalten, und er kann außerdem sichere Informa-

tionen enthalten. Somit kann Erweiterungsspeicher **674** zum Beispiel als Sicherheitsmodul für Gerät **650** bereitgestellt werden und er kann mit Anweisungen programmiert sein, die die sichere Verwendung von Gerät **650** erlauben. Zusätzlich dazu können über die SIMM-Cards sichere Anwendungen bereitgestellt werden, zusammen mit zusätzlichen Informationen, wie dem Ablegen von Identifizierungsinformationen auf der SIMM-Card auf eine Weise, die nicht gehackt werden kann.

**[0062]** Der Speicher kann beispielsweise Flash Speicher und/oder NVRAM-Speicher beinhalten, wie nachstehend erörtert. In einer Implementierung, ist ein Computerprogrammprodukt konkret in einem Informationsträger ausgeführt. Das Computerprogrammprodukt enthält Anweisungen, die, wenn sie ausgeführt werden, ein oder mehrere Verfahren ausführen, wie die oben beschriebenen. Der Informationsträger ist ein computer- oder maschinenlesbares Medium, wie der Speicher **664**, die Speichererweiterung **674** oder der Prozessorspeicher **652**, das beispielsweise über den Transceiver **668** oder die externe Schnittstelle **662** empfangen werden kann.

**[0063]** Das Gerät **650** kann drahtlos über die Verbindungsschnittstelle **666** kommunizieren, die digitale Signalverarbeitungsschaltkreise beinhalten kann, falls erforderlich. Die Verbindungsschnittstelle **666** kann Verbindungen mit verschiedenen Kommunikationstypen oder -protokollen aufbauen, darunter u. a. GSM-Sprachanrufe, SMS, EMS, oder MMS-Messaging, CDMA, TDMA, PDC, WCDMA, CDMA 2000 oder GPRS. Eine solche Kommunikation kann zum Beispiel über Funkfrequenzempfänger **668** erfolgen. Zusätzlich kann eine Kurzstreckenkommunikation stattfinden, wie unter Verwendung eines Bluetooth-, WLAN- oder anderen solchen Sende-Empfängern (nicht gezeigt). Außerdem kann GPS(Global Positioning System)-Empfängermodul **670** zusätzliche navigations- und standortbezogene drahtlose Daten für Gerät **650** bereitstellen, die ggf. von Anwendungen verwendet werden können, die auf Gerät **650** ausgeführt werden.

**[0064]** Das Gerät **650** kann mithilfe des Audio-Codecs **660** auch akustisch kommunizieren, welches gesprochene Informationen von einem Benutzer empfangen und diese in nutzbare digitale Informationen konvertieren kann. Das Audio-Codec **660** kann zudem akustische Töne für einen Benutzer erzeugen, z. B. durch einen Lautsprecher, wie beispielsweise in einem Handgerät von Gerät **650**. Diese Töne können Töne von Sprachtelefonanrufen beinhalten, können aufgezeichnete Töne (z. B. Sprachnachrichten, Musikdateien usw.) beinhalten und können auch Töne, die von Anwendungen, welche auf Gerät **650** operieren, beinhalten.

**[0065]** Das Computergerät **650** kann in einer Reihe verschiedener Formen implementiert sein, wie in der Figur gezeigt. Zum Beispiel, kann es als Mobiltelefon **680** implementiert werden. Es kann außerdem als Teil eines Smartphones **682**, Personal Digital Assistant (PDA) oder eines anderen ähnlichen Mobilgeräts implementiert sein.

**[0066]** Verschiedene Implementierungen der hier beschriebenen Systeme und Techniken können in digitalen elektronischen Schaltkreisen, integrierten Schaltkreisen, speziell konzipierten ASICs (anwendungsorientierten integrierten Schaltkreisen), Computerhardware, Firmware, Software und/oder Kombinationen davon realisiert sein. Diese verschiedenen Implementierungen können eine Implementierung in einem oder mehreren Computerprogrammen beinhalten, die auf einem programmierbaren System ausführbar und/oder interpretierbar sind, das mindestens einen programmierbaren Prozessor beinhaltet, der ein spezieller Prozessor oder ein Prozessor für allgemeine Zwecke sein kann, und der zum Empfangen von Daten und Anweisungen von und zum Übertragen von Daten und Anweisungen an ein Speichersystem, mindestens eine Eingabevorrichtung und mindestens eine Ausgabevorrichtung gekoppelt ist.

**[0067]** Diese Computerprogramme (auch bekannt als Programme, Software, Softwareanwendungen oder Code) beinhalten Maschinenanweisungen für einen programmierbaren Prozessor und können in einer höheren prozeduralen und/oder objektorientierter Programmiersprache und/oder in Assembler-/Maschinensprache implementiert werden. Wie hier verwendet, bezeichnen die Begriffe „maschinenlesbares Medium", „computerlesbares Medium" ein beliebiges Computerprogrammprodukt, eine beliebige Vorrichtung und/oder ein beliebiges Gerät (z. B. Magnetplatten, optische Platten, Speicher, programmierbare Logikbausteine (PLDs)), die verwendet werden, um einem programmierbaren Prozessor Maschinenanweisungen und/oder Daten bereitzustellen, einschließlich eines maschinenlesbaren Mediums, das Maschinenanweisungen als ein maschinenlesbares Signal empfängt. Der Begriff „maschinenlesbares Signal" bezeichnet ein beliebiges Signal, das verwendet wird, um einem programmierbaren Prozessor Maschinenanweisungen und/oder Daten bereitzustellen.

**[0068]** Zur Interaktion mit einem Benutzer können die hier beschriebenen Systeme und Techniken auf einem Computer mit einem Anzeigegerät (z. B. ein CRT-[Kathodenstrahlröhre] oder ein LCD-[Flüssigkristallanzeige]Monitor) implementiert werden, um Informationen für den Benutzer anzuzeigen, und eine Tastatur und ein Pointergerät (z. B. Maus oder Trackball), mit denen der Benutzer Eingaben in den Computer vornehmen kann. Andere Arten von Geräten können auch verwendet werden, um eine Interakti-

on mit einem Benutzer bereitzustellen; zum Beispiel kann eine dem Benutzer bereitgestellte Rückmeldung irgendeine Form von Sinnesrückmeldung sein (z. B. visuelle Rückmeldung, auditive Rückmeldung oder Tastrückmeldung); und eine Eingabe vom Benutzer kann in einer beliebigen Form empfangen werden, einschließlich akustischer, Sprach- oder Tasteingaben.

**[0069]** Die hierin beschriebenen Systeme und Techniken können in einem Computersystem implementiert sein, das eine Back-End-Komponente beinhaltet (z. B. als Datenserver), oder das eine Middleware-Komponente beinhaltet (z. B. einen Anwendungsserver), oder das eine Front-End-Komponente beinhaltet (z. B. ein Client-Computer mit einer grafischen Benutzeroberfläche oder einem Webbrowser, über welche ein Benutzer mit einer Implementierung der hier beschriebenen Systeme und Techniken interagieren kann) oder Kombination derartiger Back-End-, Middleware- und Front-End-Komponenten. Die Komponenten des Systems können durch eine beliebige Form oder ein beliebiges Medium von digitaler Datenkommunikation (z. B. ein Kommunikationsnetzwerk) miteinander verbunden sein. Beispiele von Kommunikationsnetzwerken beinhalten ein lokales Netzwerk („LAN"), ein Fernnetz („WAN") und das Internet.

**[0070]** Das Computersystem kann Clients und Server beinhalten. Ein Client und Server befinden sich im Allgemeinen ortsfern voneinander und interagieren typischerweise über ein Kommunikationsnetz. Die Beziehung zwischen Client und Server entsteht aufgrund von Computerprogrammen, die auf den jeweiligen Computer laufen und die eine Client-Server-Beziehung zueinander haben.

**[0071]** Eine Anzahl an Implementierungen wurde beschrieben. Trotzdem versteht sich, dass verschiedene Modifikationen durchgeführt werden können, ohne vom Geist und Umfang der Erfindung abzuweichen.

**[0072]** Außerdem erfordern die in den Figuren dargestellten logischen Abläufe nicht die bestimmte dargestellte Reihenfolge oder sequenzielle Reihenfolge, um wünschenswerte Ergebnisse zu erzielen. Darüber hinaus können andere Schritte vorgesehen oder Schritte aus den beschriebenen Abläufen eliminiert werden und andere Komponenten können zu den beschriebenen Systemen hinzugefügt werden oder von diesen entfernt werden. Dementsprechend liegen andere Implementierungen im Geltungsbereich der folgenden Ansprüche.

## Schutzansprüche

1. Nicht flüchtiges computerlesbares Speichermedium, das Befehle aufgezeichnet und gespeichert hat, die, wenn sie von einem Computergerät ausgeführt werden, das Computergerät zu Folgendem veranlassen:
dem Empfangen einer Anfrage einer ersten Browser-Erweiterung über eine Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen durch den Webbrowser, worin ein Empfänger der Anfrage eine zweite Browser-Erweiterung ist, ermittelt durch die Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen basierend auf einer angefragten Funktionalität, die in der Anfrage der ersten Browser-Erweiterung beinhaltet ist;
dem Senden, unter Verwendung der Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen, der Anfrage an die zweite Browser-Erweiterung, wenn die angefragte Funktionalität in der zweiten Browser-Erweiterung verfügbar ist; und
dem Senden einer Fehlermeldung an die erste Browser-Erweiterung mithilfe der Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen, wenn die angefragte Funktionalität nicht verfügbar ist.

2. Vorrichtung, umfassend:
einen Speicher; und
einen Prozessor, operativ verbunden mit dem Speicher, und so konfiguriert, dass er Code ausführt, um Folgendes zu erreichen:
dem Empfangen einer Anfrage einer ersten Browser-Erweiterung über eine Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen durch den Webbrowser, worin ein Empfänger der Anfrage eine zweite Browser-Erweiterung ist, ermittelt durch die Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen basierend auf einer angefragten Funktionalität, die in der Anfrage der ersten Browser-Erweiterung beinhaltet ist;
dem Senden, unter Verwendung der Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen, der Anfrage an die zweite Browser-Erweiterung, wenn die angefragte Funktionalität in der zweiten Browser-Erweiterung verfügbar ist; und
dem Senden einer Fehlermeldung an die erste Browser-Erweiterung mithilfe der Schnittstelle für die Programmierung von nachrichtenverarbeitenden Programmen, wenn die angefragte Funktionalität nicht verfügbar ist.

Es folgen 6 Seiten Zeichnungen

**Anhängende Zeichnungen**

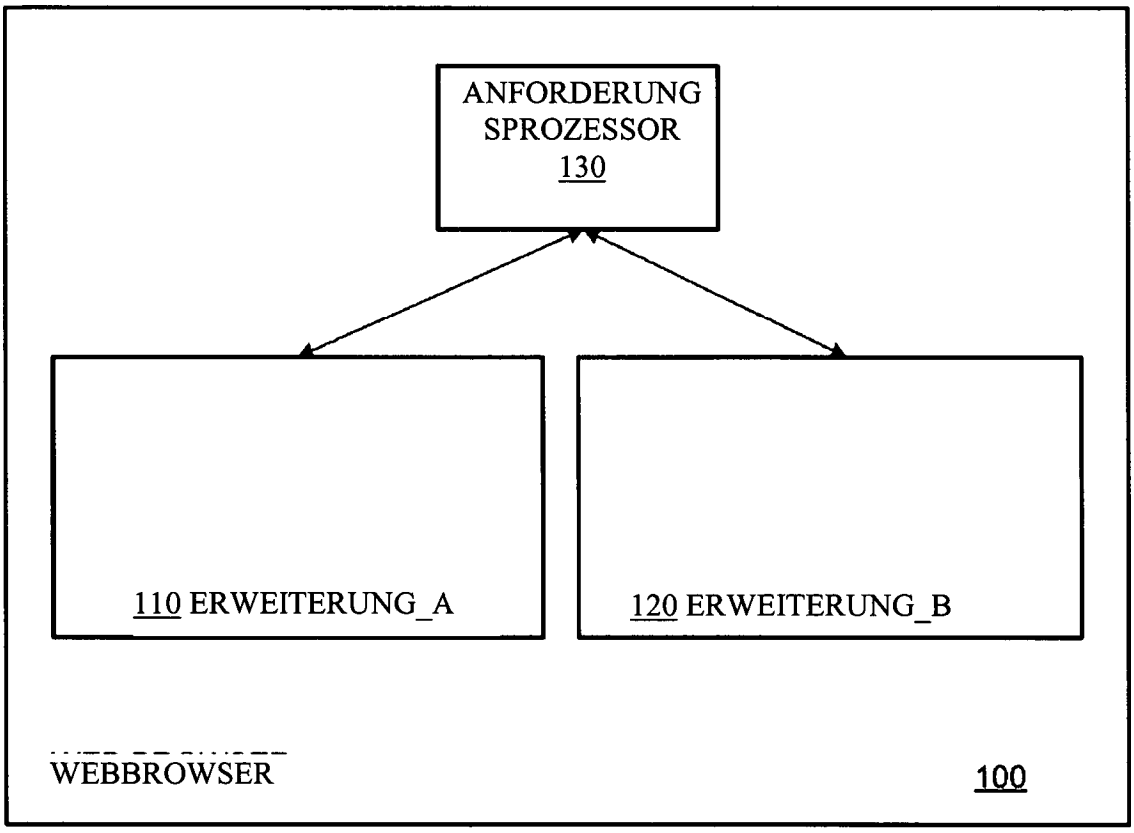



**FIG. 1**

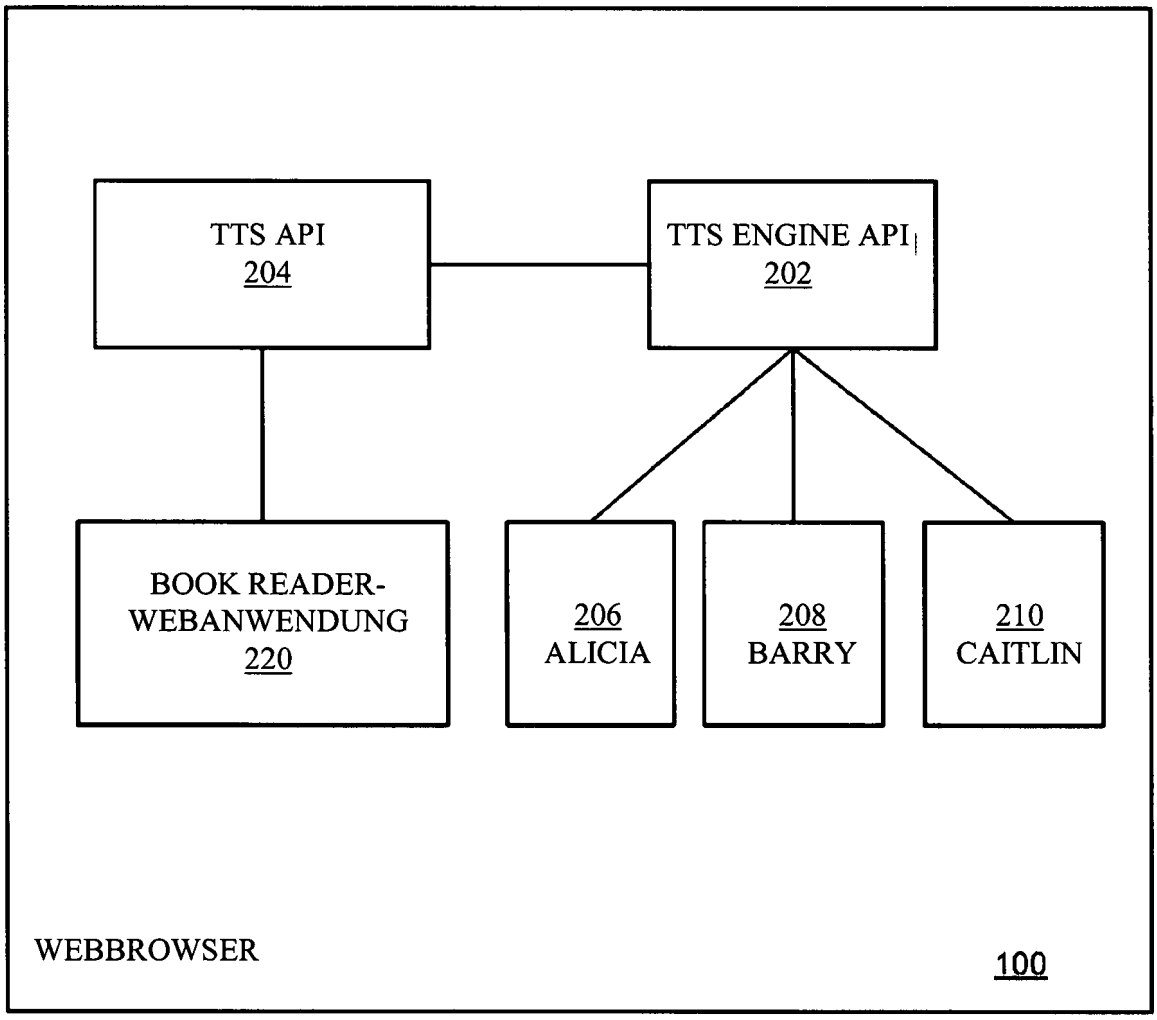

TTS API
204
TTS ENGINE API
202
BOOK READER-
WEBANWENDUNG
220
206
ALICIA
208
BARRY
210
CAITLIN
WEBBROWSER
100

**FIG. 2**

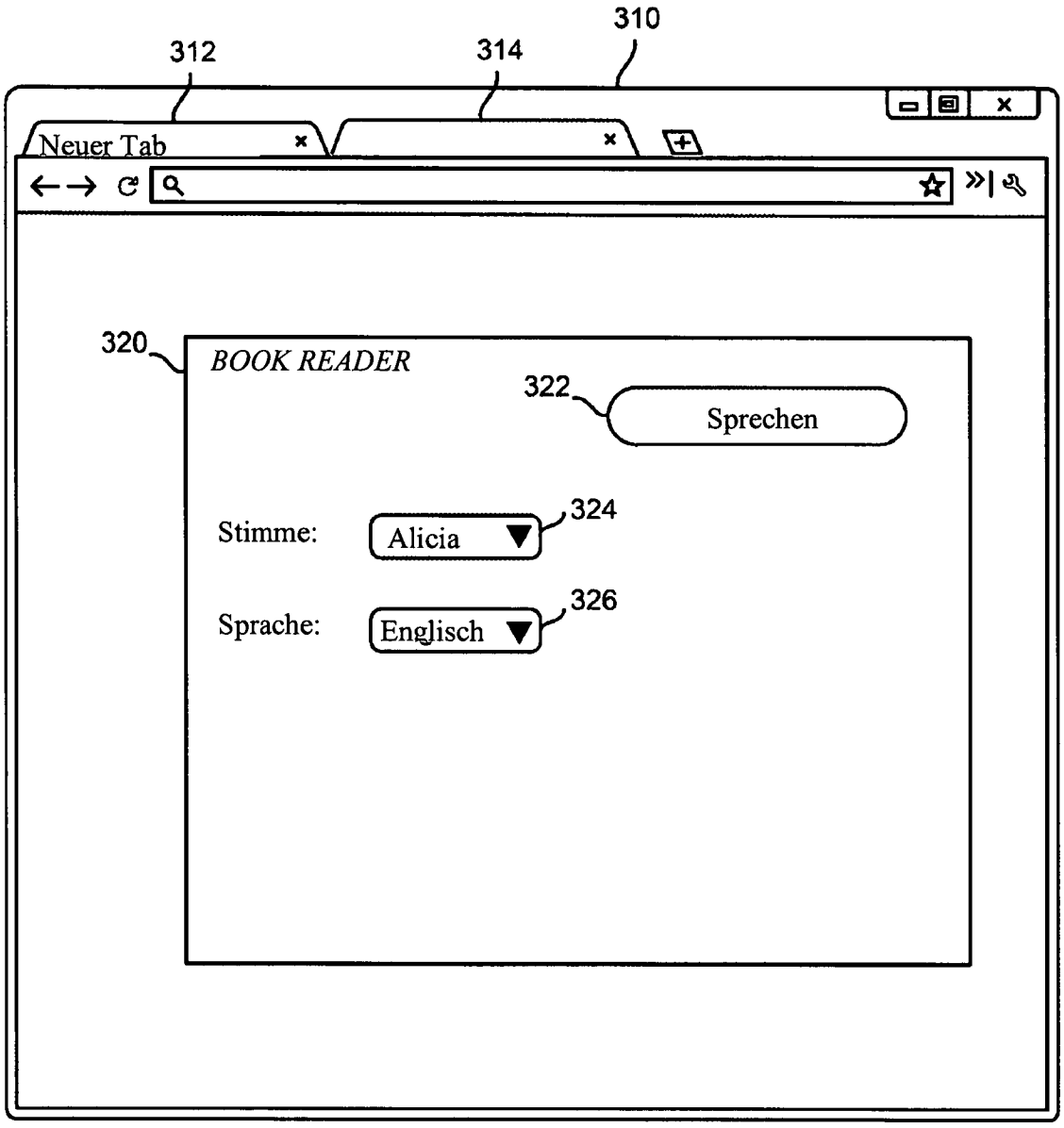

310
312
314
Neuer Tab
320
BOOK READER
322
Sprechen
Stimme:
Alicia
324
Sprache:
Englisch
326

**FIG. 3**

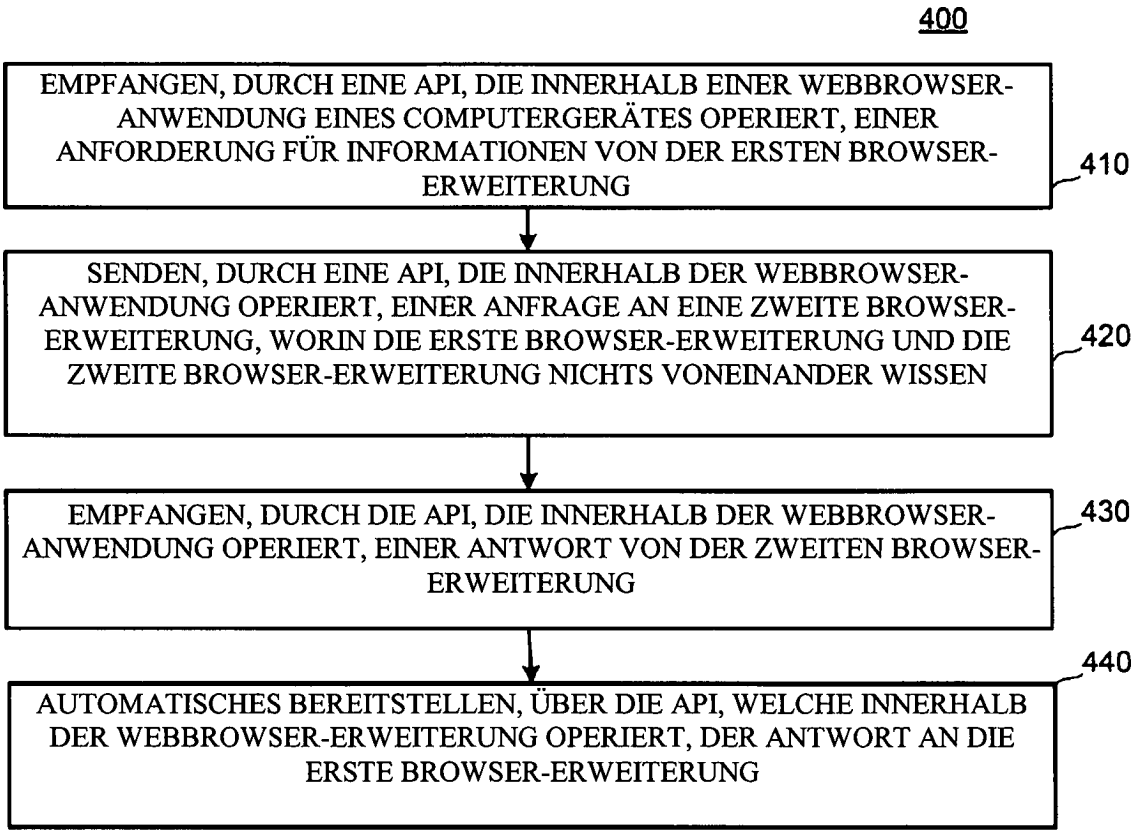

400
EMPFANGEN, DURCH EINE API, DIE INNERHALB EINER WEBBROWSER-ANWENDUNG EINES COMPUTERGERÄTES OPERIERT, EINER ANFORDERUNG FÜR INFORMATIONEN VON DER ERSTEN BROWSER-ERWEITERUNG
410
SENDEN, DURCH EINE API, DIE INNERHALB DER WEBBROWSER-ANWENDUNG OPERIERT, EINER ANFRAGE AN EINE ZWEITE BROWSER-ERWEITERUNG, WORIN DIE ERSTE BROWSER-ERWEITERUNG UND DIE ZWEITE BROWSER-ERWEITERUNG NICHTS VONEINANDER WISSEN
420
EMPFANGEN, DURCH DIE API, DIE INNERHALB DER WEBBROWSER-ANWENDUNG OPERIERT, EINER ANTWORT VON DER ZWEITEN BROWSER-ERWEITERUNG
430
AUTOMATISCHES BEREITSTELLEN, ÜBER DIE API, WELCHE INNERHALB DER WEBBROWSER-ERWEITERUNG OPERIERT, DER ANTWORT AN DIE ERSTE BROWSER-ERWEITERUNG
440

**FIG. 4**

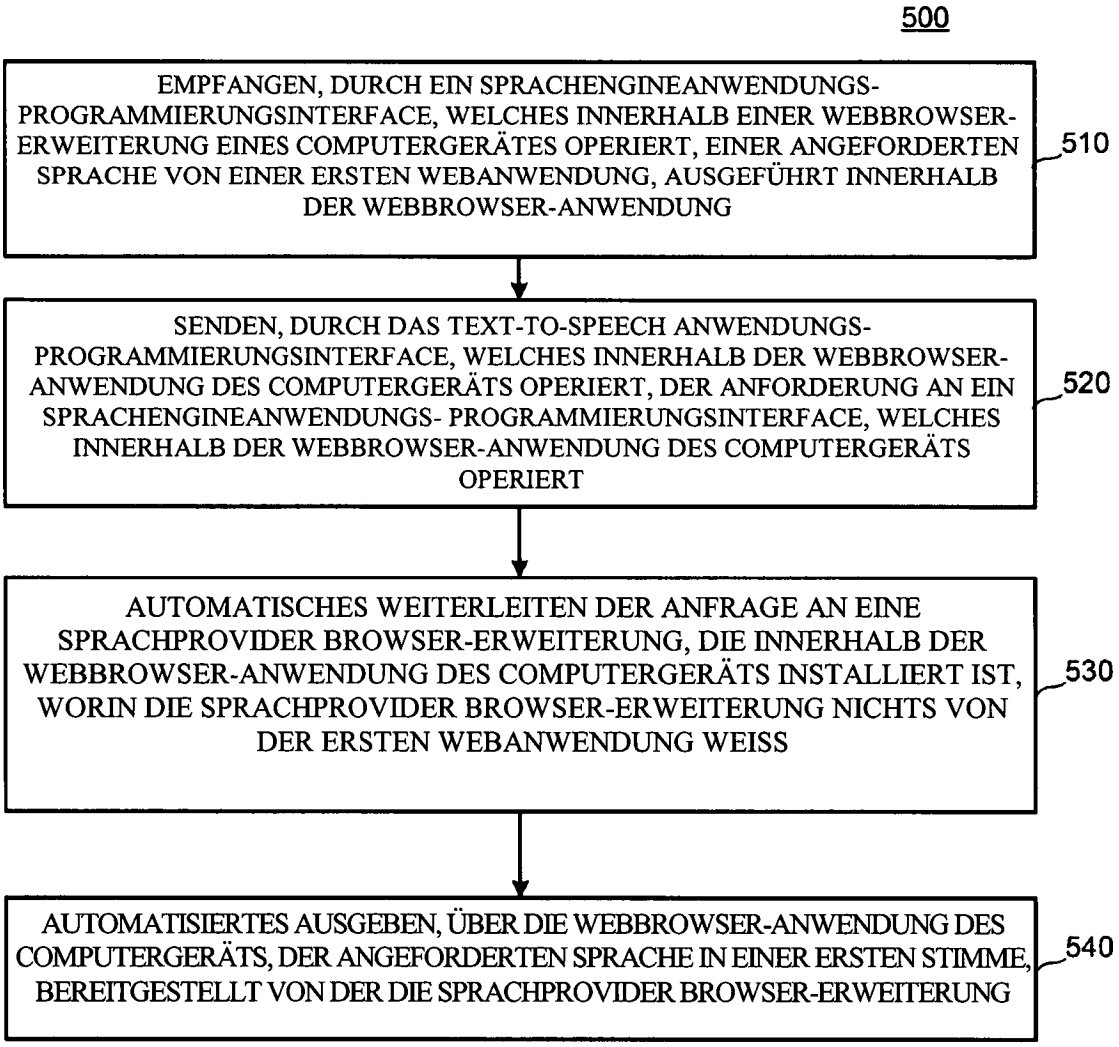

500
EMPFANGEN, DURCH EIN SPRACHENGINEANWENDUNGS-PROGRAMMIERUNGSINTERFACE, WELCHES INNERHALB EINER WEBBROWSER-ERWEITERUNG EINES COMPUTERGERÄTES OPERIERT, EINER ANGEFORDERTEN SPRACHE VON EINER ERSTEN WEBANWENDUNG, AUSGEFÜHRT INNERHALB DER WEBBROWSER-ANWENDUNG
510
SENDEN, DURCH DAS TEXT-TO-SPEECH ANWENDUNGS-PROGRAMMIERUNGSINTERFACE, WELCHES INNERHALB DER WEBBROWSER-ANWENDUNG DES COMPUTERGERÄTS OPERIERT, DER ANFORDERUNG AN EIN SPRACHENGINEANWENDUNGS- PROGRAMMIERUNGSINTERFACE, WELCHES INNERHALB DER WEBBROWSER-ANWENDUNG DES COMPUTERGERÄTS OPERIERT
520
AUTOMATISCHES WEITERLEITEN DER ANFRAGE AN EINE SPRACHPROVIDER BROWSER-ERWEITERUNG, DIE INNERHALB DER WEBBROWSER-ANWENDUNG DES COMPUTERGERÄTS INSTALLIERT IST, WORIN DIE SPRACHPROVIDER BROWSER-ERWEITERUNG NICHTS VON DER ERSTEN WEBANWENDUNG WEISS
530
AUTOMATISIERTES AUSGEBEN, ÜBER DIE WEBBROWSER-ANWENDUNG DES COMPUTERGERÄTS, DER ANGEFORDERTEN SPRACHE IN EINER ERSTEN STIMME, BEREITGESTELLT VON DER DIE SPRACHPROVIDER BROWSER-ERWEITERUNG
540

**FIG. 5**

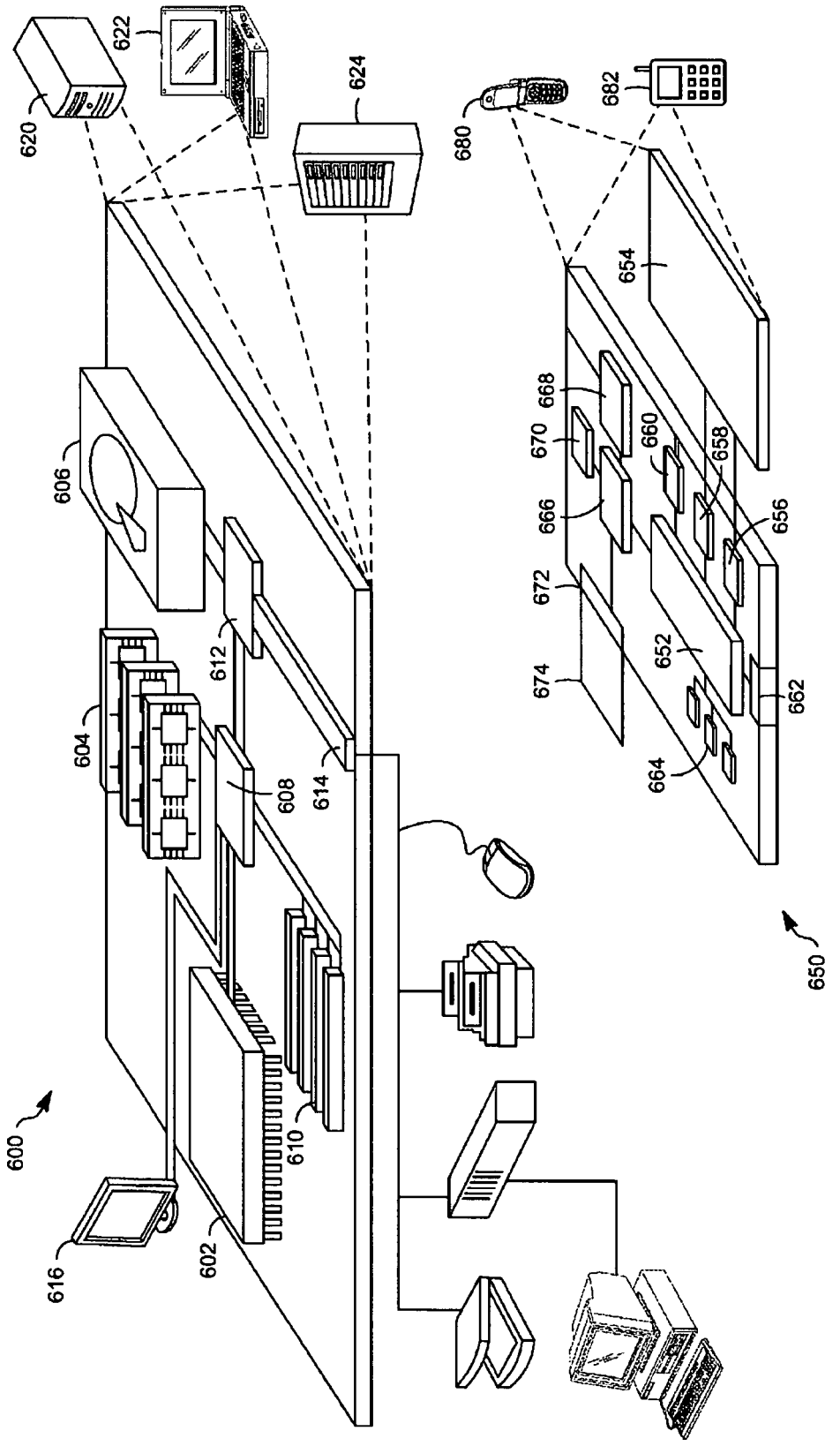

600
602
604
606
608
610
612
614
616
620
622
624
650
652
654
656
658
660
662
664
666
668
670
672
674
680
682

FIG. 6